VŠB – Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Programování mikrokontrolérů ESP32

Programming of ESP32 microcontrollers

Student: Bc. Jiří Poštulka

Vedoucí diplomové práce: doc. Ing. Marek Babiuch, Ph.D.

Ostrava 2020

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Zadání diplomové práce

Student: **Bc. Jiří Poštulka**

Studijní program: N2301 Strojní inženýrství

Studijní obor: 3902T004 Automatické řízení a inženýrská informatika

Téma: Programování mikrokontrolérů ESP32
Programming of ESP32 Microcontrollers

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Seznamte se s mikrokontroléry ESP32, vývojovým prostředím pro tvorbu aplikací a možnostmi jeho využití.
2. Navrhněte úlohu využívající mikrokontroléry ESP32 v oblasti domácí automatizace.
3. K dané úloze navrhněte vhodné hardwarové prostředky, softwarovou podporu a použité technologie.
4. Realizujte navrženou aplikaci a popište implementaci výsledného nasazení.
5. Zhodnoťte dosažené výsledky a navrhněte směr dalšího možného řešení.

Seznam doporučené odborné literatury:

Dev Center.Windows IoT: Setting up a Raspberry Pi [online], [cit. 2019-11-14]. Setting up a Raspberry Pi, 2019. *Dev Center.Windows IoT* [online]. USA: Microsoft, 22.5.2019 [cit. 2019-11-14]. Dostupné z: https://docs.microsoft.com/cs-cz/windows/iot-core/tutorials/rpi
Visual Studio Code in Action, 2019. *Visual Studio Code* [online]. USA: Microsoft, 2019 [cit. 2019-11-14]. Dostupné z: https://code.visualstudio.com/docs#vscode-in-action
ESP32 A Different IoT Power and Performance, 2019. *Espressif* [online]. Shanghai: Espressif Systems, 2019 [cit. 2019-11-14]. Dostupné z: https://www.espressif.com/en/products/hardware/esp32/resources
Python framework for multitouch apps: Kivy on Raspberry PI, 2017. *Kivy Pie* [online]. 2019 [cit. 2019-11-14]. Dostupné z: http://kivypie.mitako.eu/kivy-pie.html

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Ing. Marek Babiuch, Ph.D.**

Datum zadání: 20.12.2019

Datum odevzdání: 18.05.2020

VYSOKÁ ŠKOLA BÁŇSKÁ TECHNICKÁ UNIVERZITA Fakulta strojní -4- OSTRAVA

doc. Ing. Renata Wagnerová, Ph.D.
vedoucí katedry

prof. Ing. Ivo Hlavatý, Ph.D.
děkan fakulty

Místopřísežné prohlášení studenta

Prohlašuji, že jsem celou diplomovou práci včetně příloh vypracoval samostatně pod vedením vedoucího diplomové práce a uvedl jsem všechny použité podklady a literaturu.

V Ostravě dne: 15.5.2020

Podpis autora práce

Prohlašuji, že

- jsem byl seznámen s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo.
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen „VŠB-TUO“) má právo nevýdělečně ke své vnitřní potřebě diplomovou práci užít (§ 35 odst. 3).
- souhlasím s tím, že diplomová práce bude v elektronické podobě uložena v Ústřední knihovně VŠB-TUO k nahlédnutí. Souhlasím s tím, že údaje o kvalifikační práci budou zveřejněny v informačním systému VŠB-TUO.
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona.
- bylo sjednáno, že užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).
- beru na vědomí, že odevzdáním své práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

V Ostravě dne: 15.5.2020

Podpis autora práce

Jméno a příjmení autora práce: Bc. Jiří Poštulka

Adresa trvalého pobytu autora práce: Strmá 43/14, 747 11 Kozmice

ANOTACE DIPLOMOVÉ PRÁCE

POŠTULKA, J. *Programování mikrokontroléru ESP32: diplomová práce*. Ostrava: VŠB – Technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2020, 47 s. Vedoucí práce: doc. Ing. Marek Babiuch, Ph.D.

V této práci je představen mikrokontrolér ESP32 od společnosti Espressif. Tento mikroprocesor se stal velmi oblíbeným převážně při řešení projektů týkající se IoT. V práci jsou popsány klíčové vlastností tohoto mikroprocesoru, rovněž jsou zde uvedeny programovací prostředí a jazyky, ve kterých lze tento mikroprocesor programovat. Další částí této práce je pak návrh a realizace vlastní úlohy. Jedná se o aplikaci mikrokontroléru v oblasti domácí automatizace. Hlavním cílem úlohy, je zabezpečení domu detekováním vetřelce v objektu a jeho následném zachycení pomocí kamerového snímku. Sekundárním úkolem je sběr teplot a jejich prezentace prostřednictvím web server.

Klíčová slova: ESP32, IoT, Domovní automatizace, MQTT

ANNOTATION OF MASTER THESIS

POŠTULKA, J. *Programming the ESP32 microcontroller: master thesis*. Ostrava: VŠB – Technical University of Ostrava, Faculty of Mechanical Engineering, Department of Control Systems and Instrumentation, 2020, 47 p. Leadership: doc. Ing. Marek Babiuch, Ph.D.

In this work is presented the ESP32 microcontroller from Espressif company. This microprocessor has become very popular mainly in the IoT projects. In this thesis are described the key features of this microcontroller as well as the programming environment and languages in which this microcontroller can be programmed. Another part of this work is the design and implementation of the task itself. It is a microcontroller application in the field of home automation. The main goal of the task is to secure the house by detecting an intruder in the building and then capturing it using a camera image. The secondary task is to collect temperatures and present them through a web server.

Keywords: ESP32, IoT, Home Automation, MQTT

Obsah

1 Úvod 9

2 Softwarové prostředí pro vývoj na platformě ESP32 11

2.1 Arduino IDE 11

2.2 Espressif IoT Development Framework 12

2.3 MicroPython 14

3 Návrh úlohy využívající mikrokontroléry ESP32 v oblasti domácí automatizace. 15

4 Volba prostředků pro zaručení funkcionality úlohy 16

5 Hardwarové komponenty 17

5.1 Vstupní panel s membránovou klávesnicí 17

5.2 Vstupní panel s dotykovou obrazovkou 18

5.3 Kamerový modul 19

5.4 Řídicí jednotka 21

6 Softwarové komponenty 23

6.1 Operační systém Raspberry Pi 23

6.2 Komunikace 23

6.3 Vstupní panel 24

6.3.1 Popis zdrojového kódu 24

6.4 Kamerový modul 27

6.4.1 Popis zdrojového kódu 27

6.5 Řídicí jednotka 30

6.5.1 Databáze 30

6.5.2 Web server 30

6.5.3 Python skripty 33

7 Návrh krycích a montážních pouzder pro jednotlivá zařízení 36

7.1 Vstupní panel s membránovou klávesnicí 36

7.2 Vstupní panel s dotykovým displejem 36

8 Instalace a testování zabezpečovacího systému.....38

9 Závěr.....40

10 Bibliografie.....43

11 Seznam obrázků.....45

12 Seznam Příloh.....47

Seznam použitých zkratek a symbolů

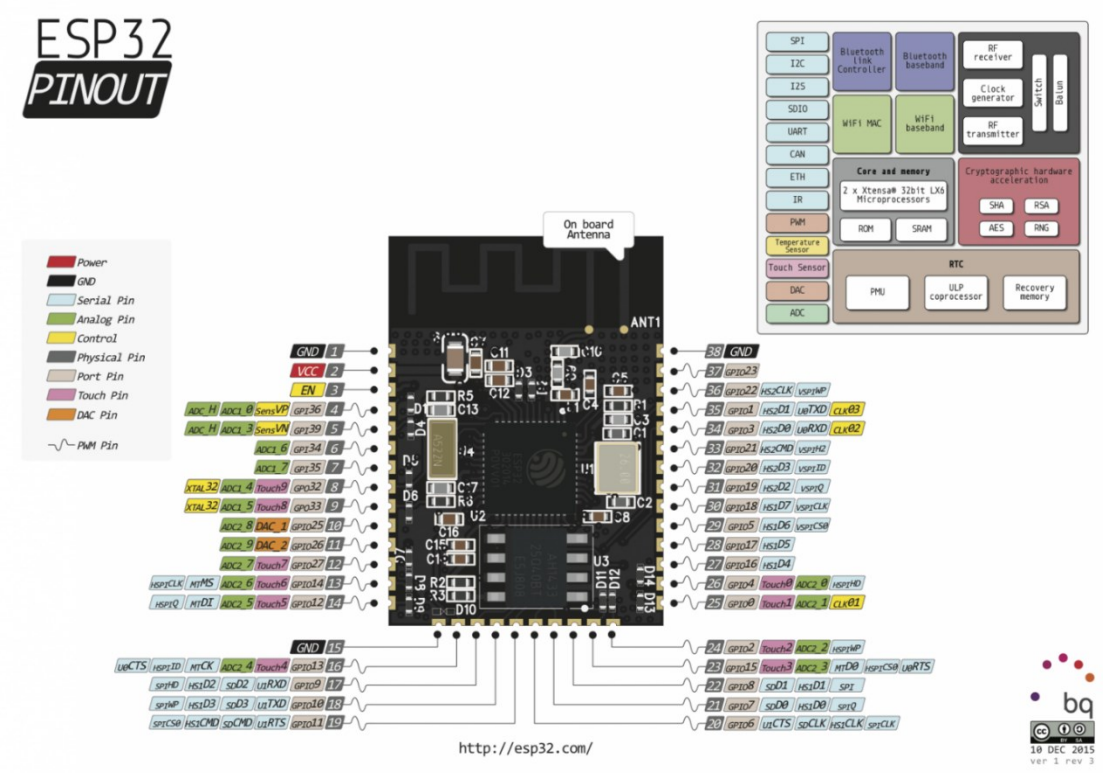
CAN	Controller Area Network
CPU	Central processing unit
GPIO	General-purpose input/output
GSM	Global System for Mobile Comunication
I/O	Input/Ouput
I2C	Inter-Integrated Circuit
I2S	Inter-IC Sound
IDE	Integrated development environment
IoT	Internet of things
IR	Infrared Radiation
LAN	Local area network
LED	light emitting diode
MAC	Media Access Control
PSRAM	Pseudo static random access memory
PWM	Pulse width modulation
SD	Secure digital
SMD	Surface mount device
SMS	Short message service
SoC	System on Chip
SPI	Serial peripheral interface
SRAM	Static random access memory
UART	Universal asynchronous receiver-transmitter
WLAN	Wireless local area network
WSL	Windows subsystem for Linux

1 Úvod

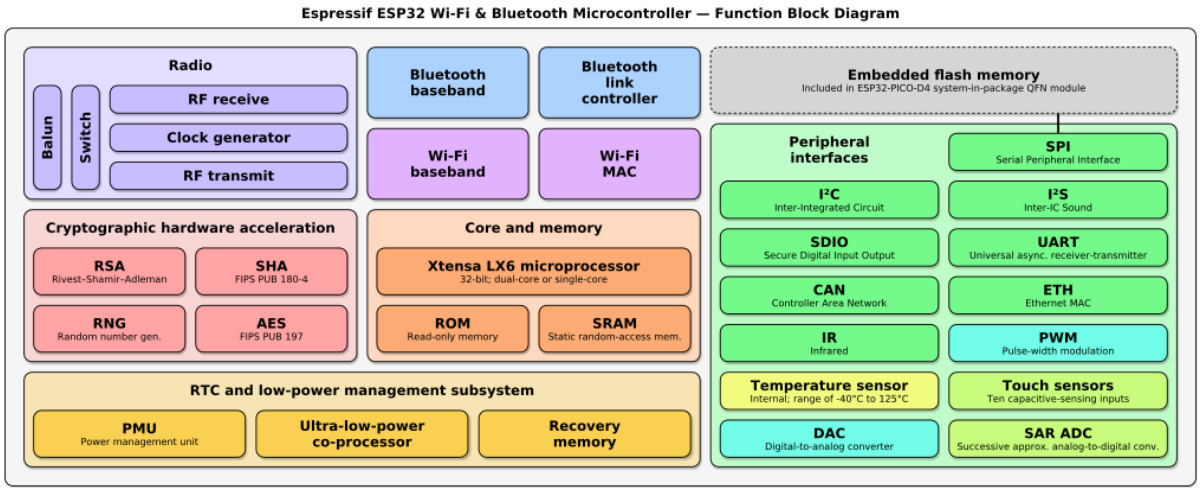
ESP32 je výkonný SoC (System on Chip) mikrokontrolér s integrovanou Wi-Fi standardu 802.11 b/g/n, dual mode Bluetooth verze 4.2 a množstvím periferií. Je pokročilým nástupcem čipu ESP8266 především v implementaci dvou jader taktovaných v různých verzích až do 240 MHz. Oproti svému předchůdci mimo zmíněných vlastností také rozšiřuje počet GPIO pinů ze 17 na 36, počet PWM kanálů na 16 a je vybaven 4MB Flash pamětí. (Stehlík, 2016)

	ESP8266	ESP32
Napětí	3,3V	3,3V
Proudová spotřeba	10uA ~ 170mA	10uA ~ 260mA
Flash	16MB max	16MB max
Procesor	Xtensa® Single-Core 32-bit L106	Xtensa® Dual-Core 32-bit LX6 600 DMIPS
Rychlost procesoru	80-160MHz	Dual 160MHz
SRAM	160kB	512kB
GPIO	17	36
A/D převodník	1x10bitový	7x12bitový
Hardware/Software PWM	0/8 kanálů	1/16 kanálů
Podpora 802.11	b/g/n/d/e/i/k/r	11b/g/n/e/i
Maximální souběžné připojení TCP	5	16
Bluetooth	-	Bluetooth 4.2
SPI/I2C/I2S/UART/CAN	2/1/2/2/0	4/2/2/2/0
Ethernet MAC interface	-	1
Pracovní teplota	-40 °C ~ 125 °C	-40 °C ~ 125 °C

(Kolban, 2016) (Kolban, 2018)



Obrázek 1 ESP32 PINOUT (Stehlík, 2016)



Obrázek 2 ESP32 Funkční blokový diagram (Function block diagram for Espressif's ESP32 series of Wi-Fi/Bluetooth chips., b.r.)

ESP32 vyvinula společnost Espressif Systems, která v současnosti poskytuje několik variant provedení ESP32 SoC ve formě ESP32 Developer Kitu, ESP32 Wrower Kitu obsahující také SD card a 3,2'' LCD display a v neposlední řadě ESP32 Azure IoT kitu s USB Bridge a dalšími vestavěnými sensory. Mimo Espressif Systems se výrobou vývojových modulů s tímto čipem zabývají také společnosti SparkFun (model ESP32 Thing DB, WeMoS se svými moduly TTGO, D1, Lolin32, Lolin D32 , Adafruid (s deskou Huzzah32), DF Robot (ESP32 FireBeeatle) a mnoho dalších, někdy více, někdy méně povedených klonů.

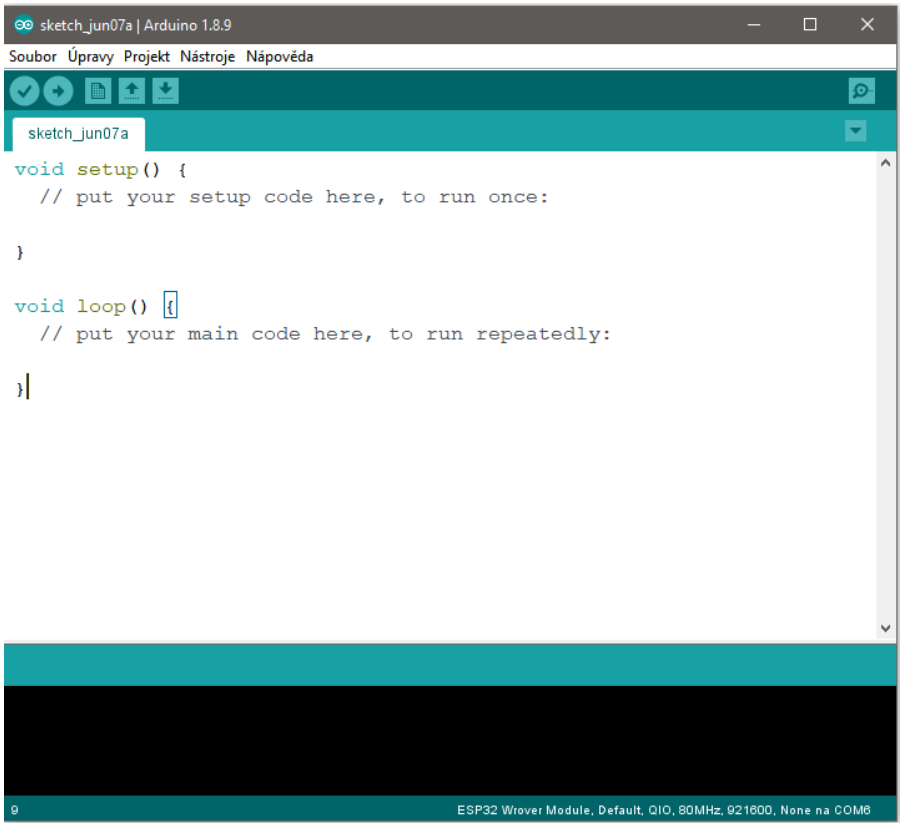
ESP32 obsahuje dvoujádrový (je možné zvolit i jednojádrovou variantu) procesor Xtensa LX6 vyrobený 40 nm technologií. Jednotlivá jádra CPU, mohou být řízená samostatně. Je k dispozici 512 KB SRAM pro data a instrukce. Z periferií máme možnost využít SPI, I2S, I2C, CAN, UART, Ethernet MAC, IR v různém množství, dle typu desky. Mezi základní vybavení patří také Hallův senzor, teplotní senzor a dotykový senzor. Další vestavěné sensory jsou implementovány ve verzích Azure IoT a Developer kitech. Desky s ESP32 jsou verzovány v provedeních, které lze prototypovat a využít v aplikacích smart home application, automation, wearebles application, audio application, IoT application s podporou cloudu a další. Je možné vybrat konkrétní development kit anebo navrhnout vlastní embedded systém postavený na mikrokontroléru ESP32. (Espressif Systems, b.r.)

2 Softwarové prostředí pro vývoj na platformě ESP32

Vývoj na platformě ESP32 nám dává mnoho možností použití. V této kapitole si popíšeme některá základní nastavení a konfigurace vývojových prostředí pro platformu ESP32. Na ESP32 zařízení, lze vyvíjet z libovolného operačního systému Windows, Linux nebo macOS. Mikrokontroléry ESP32 lze programovat pomocí různých jazyků a specifických vývojových prostředí, které nám dávají určité výhody nebo omezení.

2.1 Arduino IDE

Nejjednodušší způsob, jak začít psát kód pro ESP32 platformu, je využití Arduina platformy. Arduino platforma je open-source platforma určená pro rychlou tvorbu prototypů založených na mikrokontrolérech Atmel. Vývoj programů pro mikrokontroléry se vytváří v Arduino IDE, který je napsaný v jazyce Java. Jedná se o software vzniklý z výukového prostředí Processing, které bylo mírně upraveno a rozšířeno o určité funkce jako podpora jazyka Wiring. Wiring je programovací jazyk vytvořený pro programování mikrokontroléru bez specifických znalostí hardwaru a má podobu frameworku v jazyce C++. Wiring vyžaduje mikrokontrolér se zaváděcím programem. V dnešní době Arduino IDE nepodporuje pouze mikrokontroléry od společnosti Atmel, ale lze mu doinstalovat i jiné typy mikrokontroléru. Společnost Espressif se proto rozhodla využít této možnosti a vytvořila plugin nazvaný Arduino core for ESP32 Wi-Fi chip, který instaluje podporu pro vývoj na mikrokontroléry ESP32 do Arduino IDE prostředí. Rozšíření Arduino core for ESP32 Wi-Fi chip je uvolněno jako open-source projekt. Protože ESP32 Chip je nativně dvoujádrový, je do rozšíření Arduino core for ESP32 Wi-Fi chip implementována podpora FreeRTOS systému, které nám umožní taskové zpracování úlohy. Zároveň nám i přináší podporu optimalizace zpracování konkrétních tasků na konkrétním jádře ESP32 chipu. Standardní program napsaný pomocí Arduino platformy je rozdělen do funkce *setup*, kde se provádí inicializace hodnot při staru mikrokontroléru a do funkce loop, která je zodpovědná za běh programu. Při vývoji na ESP32 mikrokontroléry je funkce loop vždy spouštěna na prvním jádře ESP32 chipu. Musíme si však uvědomit, že číslování jader na Chipu ESP32 začíná od nultého indexu.



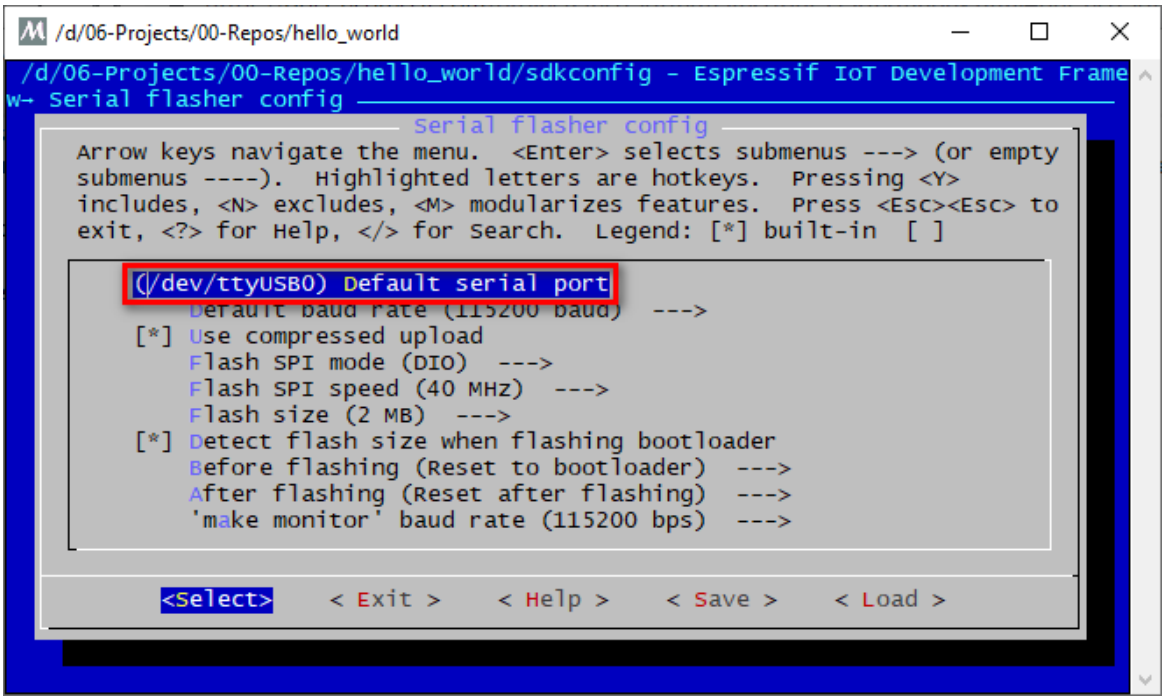
Obrázek 3 Arduino IDE

Výhody využití tohoto prostředí je jeho jednoduchá konfigurace a dobrá komunitní podpora. Za použití tohoto prostředí se dá velmi rychle začít s prototypováním na mikrokontroléru ESP32. Toto prostředí je určeno pro vývoj jednodušších projektů, při složitých projektech dochází ke ztrátě přehlednosti a udržitelnosti projektu. Zároveň je tato platforma ideální pro získání prvotních zkušeností s vývojem na mikrokontroléru ESP32. (Arduino IDE Tutorials, b.r.)

2.2 Espressif IoT Development Framework

Pokud chceme vyvíjet složitější a optimalizovanější embedded systémy postavené na mikrokontroléru ESP32, je vhodné zvolit Espressif IoT Development Framework, který nám umožní vyvíjet aplikace nativním způsobem. Espressif IoT Development Framework je open source projekt. Tento framework obsahuje možnosti základní konfigurace pro desku ESP, kompilaci zdrojového kódu a firmware downloader pro ESP32 desky. Pro vývoj aplikací se používá jazyk C. Tento framework je sice multiplatformní, ale nativně je vyvíjen pod operačním systémem Linux, kde se dá nejjednodušeji konfigurovat. Pro použití frameworku na operačním systému Windows je nutné využít doinstalovat podporu emulace UNIX-like / POSIX prostředí, nebo využít možnosti WSL, která se nachází jako součást systému Windows 10. Tak jako Arduino core for ESP32 Wi-Fi chip rozšíření do Arduina platformy i IoT Development

Framework implementuje podporu FreeRTOS systému. Vývoj s Espressif IoT Development Framework vyžaduje dodržení některých postupů, jinak výsledný projekt nebude kompilovatelný. Každý projekt musí obsahovat soubor *sdkconfig*, který slouží k ukládání informací o konfiguraci frameworku pro daný projekt. Rozhraní pro konfiguraci projektu se vyvolá vykonáním příkazu *make menuconfig*. Přes toto rozhraní se například konfiguruje, na kterém portu se dane zařízení nachází, na jakých frekvencích mají jednotlivé sběrnice pracovat, nebo zda se pro program využívá FreeRTOS v konfiguraci jednoho nebo dvou jader chipu ESP32. Tyto údaje jsou důležité, protože generují závislosti při build procesu na základě nakonfigurovaného projektu.



Obrázek 4 Espressif IoT Development Framework

Dalším krokem je pak samotný build programu, který se vyvolá příkazem *make build*. Pokud je po buildu vše v pořádku, tak je možné daný zkompilovaný program nahrát do ESP32 mikrokontroléru. K tomuto slouží vestavěný nástroj pro flashování firmwaru, který se vyvolá pomocí příkazu *make flash*.

Pro monitorovaní aplikace je možné použít vestavěný monitor, který se spustí pomocí příkazu *make monitor*.

Velkou výhodou využití tohoto frameworku je, že lze vytvářet aplikace na nejnižší úrovni. Proto se tento framework hodí na vývoj rozsáhlých a složitých projektů. Nevýhodou využití frameworku jsou poměrně vysoké počáteční zkušenosti kladené na vývojáře, který embedded systém bude vyvíjet. Komunita kolem tohoto frameworku také není zdaleka tak rozsáhlá jako v případě Arduino platformy.

2.3 MicroPython

Někteří výrobci, kteří vyrábějí vývojové desky založené na chipu ESP32 jako defaultní firmware, nyní používají Micropython prostředí. Nejčastěji se s tím setkáme na chipech ESP32-Wrover, které obvykle mají k dispozici i PSRAM. Micropython je odlehčená implementace programovacího jazyka Python 3, která obsahuje podmnožinu standardních knihoven Pythonu a je optimalizovaná pro běh na mikrokontrolérech. Výhoda Pythonu jako programovacího jazyku je, že se dá poměrně rychle naučit.

Každé řešení by mělo obsahovat minimálně dva soubory a to *boot.py* a *main.py*. Odpovídá to konceptu použitého v Arduino platformě. V souboru *boot.py* se definují inicializace proměnných, které se nastaví při spuštění mikrokontroléru a main.py obsahuje program pro mikrokontrolér, který se vykonává neprodleně po inicializaci proměnných v *boot.py* souboru. Pokud je potřeba využít dalších knihovních funkcí, které nejsou implementovány v základním firmwaru micropythonu, musí se tyto knihovní funkce na zařízení flashnout také. Zpravidla se jedná o knihovny zabezpečující komunikaci a chování konkrétního hardwarového prvku, např. displej, vibrační senzor atd. Nahrání programu a knihovních funkcí do mikrokontroléru lze provést pomocí python utility ampy, která zabezpečuje komunikace s mikrokontrolérem přes sériovou linku. Přes tuto utilitu se provádí veškerá manipulace se soubory na mikrokontroléru. Příkaz pro nahraní *main.py* souboru na mikrokontrolér vypadá:

```
ampy -p <USB-to-Serial Port> put main.py
```

Pro debuggování Pyhonu na mikrokontroléru lze použít terminálové připojení, například pomocí programu PuTTY. Pro psaní python programů lze využít jakýkoli dostupný textový editor, např. Visual studio Code.

Tento přístup vývoje je výhodný pro prototypování a vývoj nových algoritmických řešení pro embedded systémy postavené na firmware Micropythonu. Náročnost vývoje je poněkud vyšší než v případě Arduino platformy, protože neexistuje jednoduché rozhraní, ale je to zabezpečeno množinou command line utilit.

Pro mikrokontroléry ESP32 existují ještě další způsoby, jak na ně vyvíjet aplikace, například použitím programovacího jazyka Lua a příslušného firmwaru nebo využitím programovacího jazyka JavaScript. Další možností je využít některý z komerčních projektů např. Zerynth, které podporují mikrokontroléry ESP32.

3 Návrh úlohy využívající mikrokontroléry ESP32 v oblasti domácí automatizace.

Odlišností oproti předchůdci ESP32 nebo podobným mikrokontrolérům, jako jsou například desky Arduino, je dvoujádrový procesor. Proto jsem se rozhodl, že se pokusím naplno využít potenciál tohoto mikrokontroléru. Abych tak učinil, bylo nutné nalézt proces, který mikrokontrolér zatíží co možná nejvíce a současně bude tento proces využitelný v oblasti domovní automatizace. Tato úvaha mě vedla k použití kamery, proto jsem se rozhodl jí zakomponovat do této úlohy.

Při spojení kamery v oblasti domovní automatizace se nabízí možnost využití v oblasti zabezpečovací techniky.

V dnešní době je velmi oblíbené ve spojitosti s technologiemi používaní slova „smart“. Aby si i toto zařízení zasloužilo tento přívlastek, bylo nutné takto navrhnutou sestavu o něco obohatit, jinak by se jednalo pouze o „hloupou“ kameru, která by v nekonečných smyčkách natáčela záznam a ukládala jej na SD kartu.

Využitím pasivního infračerveného čidla (dále PIR čidlo) vznikne sestava, která bude vědět, že ve sledované oblasti nastal pohyb a je nutné začít tuto oblast zaznamenávat. Mohlo by však docházet k zachycení osob, které se běžně po těchto prostorách pohybují, což je nežádoucí. Proto je třeba, aby zařízení vědělo, že zaznamenaný pohyb je právě ten, který chceme zachytit kamerou.

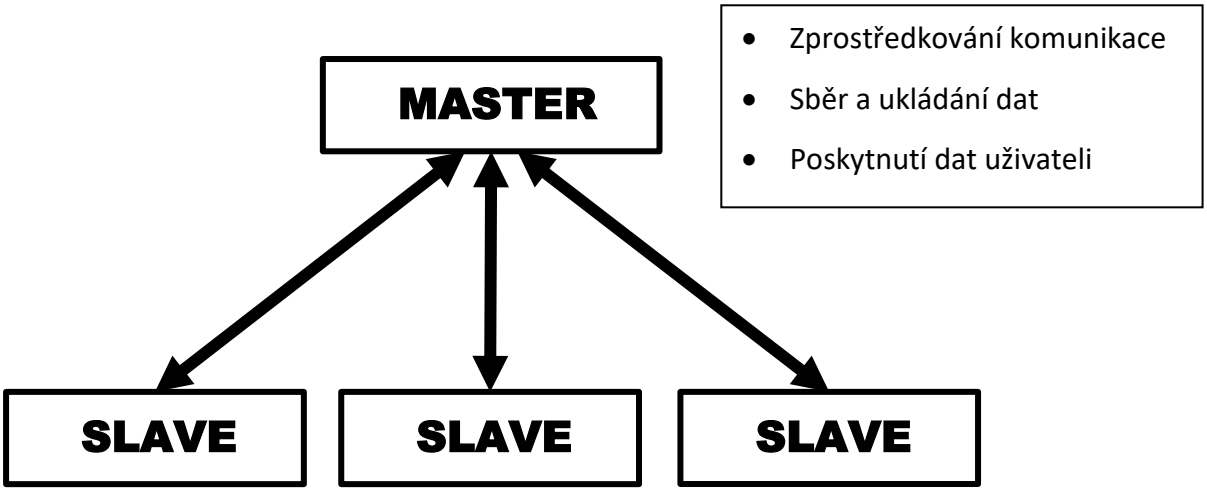
To nás spojuje s dalším, velmi hojně používaným slovem, kterým je IoT neboli internet věcí. Nutným předpokladem pro takovéto zařízení je právě zmíněná komunikace s okolním světem. K tomu jsem využil bezdrátové Wi-Fi komunikace, která už je přímo implementována na ESP32. Díky této komunikace bude zajištěno, že se zařízení dozví, kdy je správný okamžik sledovat jemu přidělenou plochu, došlo-li k nějakému pohybu v této oblasti a pokud tomu tak je, tak aby začal pořizovat kamerový záznam.

Abychom mikrokontrolér využívali i v době, kdy nebude potřeba pořizovat kamerový záznam, rozhodl jsem se k němu přidat i teplotní čidlo, které bude v určitém intervalu zaznamenávat teplotu pro lepší přehled ohledně vytápění domu.

4 Volba prostředků pro zaručení funkcionality úlohy

Navrhl jsem zabezpečovací systém domu, který bude současně s pořizováním kamerových snímků sbírat i teplotu. Aby bylo možné kamerové snímky nebo teplotu zobrazovat, je nutné, aby byla všechna tato data pohromadě. Při návrhu architektury jsem se rozhodl použít typ MASTER-SLAVE. Přičemž master by zprostředkovával komunikaci mezi jednotlivými moduly a současně by sbíral, ukládal a zobrazoval veškerá důležitá data *viz Obrázek 5*.

Z obrázku je patrné, že master bude obsluhovat všechna zařízení v této úloze. Proto je nutné, aby měl dostatečný výpočetní výkon. Z toho důvodu jsem se rozhodl použít Raspberry Pi, coby řídicí jednotku (master).



Obrázek 5 Architektura navržené úlohy

Raspberry Pi

Raspberry Pi je jednodeskový mikropočítač o velikosti platební karty. Jádrem zařízení je ARM procesor od firmy Broadcom. Jde o takzvaný SoC, System on a Chip, kdy jediný čip obsahuje prakticky veškerou funkcionalitu a vyžaduje jenom minimum podpůrných obvodů. Tuto architekturu využívá většina současných smartphonů a tabletů. Obsahuje standardní rozhraní typu USB, Ethernet, HDMI a jeden univerzální GPIO (General Purpose Input and Output) konektor. Lze na něj nainstalovat jakýkoliv operační systém kompatibilní s ARM platformou. Typicky se bude jednat o různé varianty Linuxu či Androidu, ale existuje třeba i speciální edice Windows 10 for IoT.

Autoři odhadují, že přibližně třetina těchto počítačů se používá pro výrobu, třetina v rámci různých vestavěných průmyslových zařízení a třetinu používají domácí „bastlíři“ pro své vlastní projekty. (Valášek, b.r.)

5 Hardwarové komponenty

Sestava tedy bude obsahovat Raspberry Pi, coby řídicí jednotku. Dále budou v sestavě obsaženy dva kamerové moduly vybavené PIR a teplotním čidlem. Pro uzamčení domu bude sloužit ESP32 osazený displejem a membránovou klávesnicí pro vstup do kotelny a dotykový displej rovněž řízený ESP32 u hlavního vstupu. Součástí sestavy je i jeden mikroprocesor ESP32 osazený dvěma teplotními čidly, stejnými jako má kamerový modul, proto toto zařízení nebudu v následující kapitole nijak popisovat.

5.1 Vstupní panel s membránovou klávesnicí

Sestava obsahuje následující prvky:

- ESP-WROOM-32
- jednobarevný znakový 16x2(16 znaků na řádek a 2 řádky) LCD displej
- eses membránová klávesnice 4x4 pro jednodeskové počítače

ESP-WROOM-32

Jedná se o Wi-Fi + BT MCU modul, který je vhodný pro širokou škálu aplikací, od nízkonapěťových senzorů až po náročnější úkoly, jako je komprese hlasového záznamu, streamování hudby nebo dekódování MP3. Jádrem tohoto produktu je čip ESP32-D0WDQ.

ESP32 integruje bohatou sadu periferií, od kapacitních dotykových snímačů, Hallových snímačů, zesilovačů s nízkým šumem, rozhraní SD karet, Ethernet, SPI, UART, I2S a I2C.Integrace Bluetooth, Bluetooth LE a Wi-Fi zajistí, že široký rozsah aplikací může být cílen.

ESP32 podporuje přenosovou rychlost až 150Mbps a 20.5 dBm výstupní výkon antény.

LCD displej

Tento displej (*viz Obrázek 7*) je připojen přes I2C sběrnici. Jedná se o sériovou sběrnici, která připojené zařízení rozděluje do kategorie master nebo slave. Sběrnice je připojena pouze pomocí dvou vodičů. Jeden vodič slouží k přenosu hodinového signálu (SCL – synchronous clock) a je datový kanál (SDA – synchronous data).

Obrázek 7 (ESP-32 Wroom, b.r.)

Obrázek 6 (Arduino LCD displej, 2017)

Membránová klávesnice

4x4 Klávesnice obsahuje znaky (1÷9, A÷D a speciální znaky # a *), ty jsou připojeny na 8 pinů. Neboť tlačítka v jednotlivých řádcích a sloupcích jsou vždy vyvedena na společný drát. Po stisknutí tlačítka se vždy v daném bodě spojí řádek a sloupec. Výrobce garantuje životnost ve výši 100 milionů stisknutí.

Pasivní reproduktor

Poslední komponentou v této sestavě je alarm, jehož pracovní napětí je v rozsahu 3-24V a intenzita 95dB. Jeho funkcí je v případě vniknutí vetřelce do domu spustit hlasitý tón, který by měl upozornit na vniknutí do domu.

Obrázek 9 (Membránová klávesnice, b.r.)

Obrázek 8 (Alarm, b.r.)

5.2 Vstupní panel s dotykovou obrazovkou

U hlavního vstupu pak bude stejný model ESP32 jako u předchozí sestavy. Co se zobrazovací jednotky týče, bude použit dotykový 2,8” USART displej od společnosti NEXTION. Největší výhodou těchto displejů je grafický editor, ve kterém se dá snadno a bez problému

nadefinovat grafické a dotykové prostředí a provést simulace. Díky tomuto editoru se při vývoji aplikací ušetří spousta času. Rozlišení tohoto displeje je 320x240 spolu s 65 tisící barvami a nastavitelným jasem se postarají o dokonalou volbu pro všechny druhy aplikací. Největší předností je komunikace prostřednictvím USART, přes kterou lze nahrát vytvořené grafické prostředí přímo do displeje, ve kterém je integrovaná 4MB flash paměť. Druhou variantou použití micro SD karty, pro kterou má tento displej slot. USART rozhraní rovněž slouží ke komunikaci s mikroprocesorem, prostřednictvím něho přijímá proměnné a odesílá odezvu při klinutí na prvek. Napájecí napětí odpovídá hodnotě 5 V s odběrem 90mA.

Obrázek 10 (2,8" USART displej od společnosti NEXTION, b.r.)

5.3 Kamerový modul

Sestava obsahuje následující prvky:

- Ai-Thinker ESP32-CAM
- Teplotní čidlo DALLAS DS18B20
- PIR modul

Ai-Thinker ESP32-CAM

Opět se jedná o Wi-Fi + BT MCU modul, rozšířený o 520KB SRAM včetně externí 4MPSRAM. Dále podporuje OV2640 a OV7670 kamery a SD karty. ESP32-CAM může být široce používán v různých IoT aplikacích. Je vhodný pro domácí inteligentní zařízení, průmyslové bezdrátové ovládání, bezdrátové monitorování a další aplikace IoT.

Obrázek 11 (Ai-Thinker ESP32-CAM, b.r.)

RGB led

Jedná se o typ RGB led se společnou anodou, tedy že jsou jednotlivé ledky spínany připojením k zemi. Její funkcí bude informace o aktuálním stavu kamerového modulu (jestli je připojen k Wi-Fi síti, nebo jestli nedošlo k chybě s načtením kamery či jiné oznámení o chybě) indikací v různých barvách.

Teplotní čidlo DALLAS DS18B20

Toto čidlo umožňuje měřit teplotu v rozsahu -55 až +125 stupňů Celsia, přičemž v rozsahu -10 až +85 stupňů Celsia má garantovanou přesnost ±0,5 °C. Je dostupný v pouzdře TO-92, které se velikostí podobá obyčejným tranzistorům, nabízí se i ve vodotěsné variantě, kdy je senzor zataven do nerezové tyčinky. Pro komunikaci je využita sběrnice OneWire, která využívá pouze jeden komunikační pin. Toto čidlo také podporuje takzvaný parazitní režim, kdy pro spojení čidla mikrokontrolérem stačí využít pouze dva vodiče. (Maxim Integrated DS18B20, b.r.)

PIR modul

Je mnoho variant pohybových senzorů určených pro podobné aplikace jako je tato. Já opět vybíral ze dvou variant. První z nich nese označení HC – SR501, jehož napájecí napětí je v rozsahu 4,5 až 20 V a výstupní logika je 0/3,3V. Rozměr snímače je 32x24mm, je to tudíž vetší čidlo, ovšem umožňuje pomocí dvou potenciometrů nastavit citlivost snímače a časování. Rovněž i jeho detekční vzdálenost je více než dostačující, výrobce uvádí vzdálenost do 7 metrů se snímaným úhlem 120°.

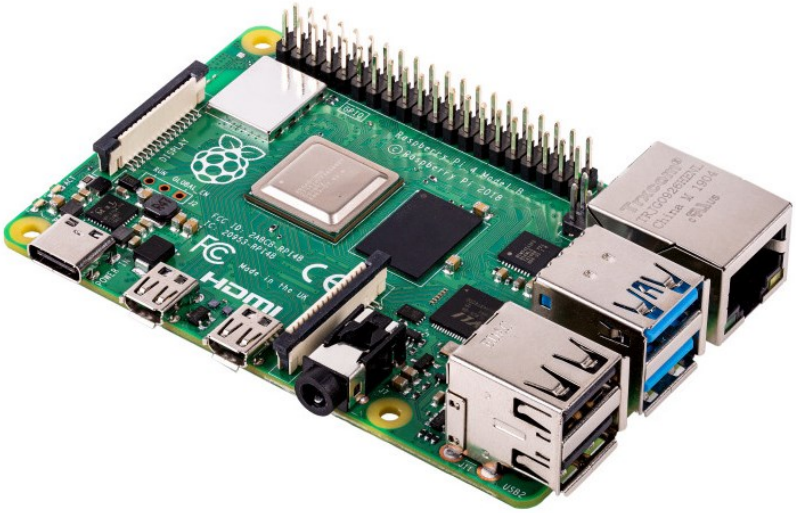
Druhou variantou je menší modul s označením AM312. Pracovní napětí má v rozsahu 2,7 až 12 V, přičemž výstupní logika je stejná jako u předchozího modulu. Rozměry destičky je pouhých 10x8mm. Ovšem i detekční vzdálenost je menší, přibližně do rozsahu 3-5m při snímaném úhlu 110°.

Obrázek 12 (PIR modul HC-SR501, b.r.)

Obrázek 13 (PIR modul AM312, b.r.)

5.4 Řídicí jednotka

Jelikož je už 4. generace Raspberry Pi, tak jsem volil mezi právě tímto nejnovějším modelem a starším modelem Zero W, které není příliš výkonné, ale pro tuto sestavu dostačující.



Obrázek 14 (Raspberry Pi 4 Model B, b.r.)

Raspberry Pi 4 je nabízen ve variantách dle velikosti operační paměti, tj. od 1, 2 nebo 4 GB RAM. Procesor ARM Cortex-A72 čítá 4 jádra s taktem 1,5GHz. Přechod na modernější 28nm technologii umožnil Raspberry Pi 4 B dosáhnout významného zvýšení výkonu procesoru, multimédií a I/O, díky čemuž je nyní plnohodnotným osobním počítačem. Napájecí konektor byl pozměněn na USB-C, díky tomu je umožněn vyšší výkon. Pro připojení monitoru zde slouží dva microHDMI konektory.

Dále obsahuje dva USB 3.0 porty a dva USB 2.0 porty. Připojení k Wi-Fi nebo 1Gbit/s ethernet konektoru, stejně tak nechybí ani Bluetooth verze 5.0.

Raspberry Pi Zero W je levná, úsporná a zmenšená varianta populárního Raspberry Pi. Zero W má navíc vestavěnou Wi-Fi a Bluetooth. Zero je postaveno s jednojádrovým procesorem ARMv6 taktovaným na 1 GHz. Dále disponuje 512 MB operační paměti, audiovizuálním výstupem přes Mini-HDMI, klasickým microUSB 2.0 konektorem a napájením přes microUSB. K dispozici má navíc 40 pinové GPIO rozhraní.

Řídicí jednotka je pak dále rozšířena o GSM modul s označením IOT-GA6-B. Jedná se o ideální řešení pro IoT zařízení, jenž komunikuje prostřednictvím sériové linky na TTL úrovni. Podporuje hlasové hovory, SMS, přenos dat GPRS, standardní AT příkazy.

Samozřejmě by bylo možné připojit tento modul k mikroprocesoru ESP32, ale vzhledem k tomu, že by bylo nutné veškeré zprávy přeposílat z řídicí jednotky na vstupní panel a pak až ke koncovému zařízení, tak by byla tato realizace příliš komplikovaná. Z tohoto důvodu jsem se rozhodl jednoho prostředníka vynechat a zařízení připojit přímo k řídicí jednotce.

Obrázek 15 (Raspberry Pi Zero W, b.r.)

Obrázek 16 (GSM modul, b.r.)

6 Softwarové komponenty

V této kapitole jsou popsány jednotlivé služby, které byly v této sestavě použity. Nachází se zde i popisy fragmentů zdrojových kódů, které jsou nahrány na jednotlivých zařízeních. Díky tomuto nastínění lze lépe pochopit způsob fungování celé sestavy.

6.1 Operační systém Raspberry Pi

Zvolil jsem Raspbian, jehož nejnovější verze nese název Buster a běží na verzi Debianu Linux 4.16 (Wostl, b.r.). Jedná se o operační systém navržený přímo od výrobce této desky, díky tomu by měla být zaručená spolehlivost a stabilita tohoto systému. Je nabízena v desktopové verzi s doporučenými softwary, v čisté desktopové verzi anebo ve verzi lite, jenž je minimální verze pouze s příkazovou řádkou, bez grafického prostředí (Raspbian, b.r.). Pro moji úlohu jsem zvolil verzi lite, protože Zero není dostatečně výkonné, aby zvládlo desktopovou verzi systému.

Nahráno na Raspberry Pi:

MQTT Server - jedná se o službu sloužící pro komunikaci s mikrokontroléry.

Apache 2 - služba poskytující web server.

MariaDB - databáze na kterou jsou ukládány např. teploty z čidel.

Skripty v Pythonu - zastřešují veškerou logiku při komunikaci s mikrokontroléry, jakožto i ukládání dat do databáze.

6.2 Komunikace

Veškerá komunikace bude probíhat bezdrátově prostřednictvím Wi-Fi. Protokol, kterým budou jednotlivé zařízení komunikovat, bude MQTT.

MQTT (dříve: Message Queuing Telemetry Transport, dnes MQ Telemetry Transport) byl uveden na svět v roce 1999 společností IBM. Jedná se o jednoduchý a nenáročný protokol pro předávání zpráv mezi klienty prostřednictvím centrálního bodu – brokeru. Díky této nenáročnosti a jednoduchosti je snadno implementovatelný i do zařízení s „malými“ procesory a poměrně rychle se rozšířil. (Waher, 2015)

U protokolu MQTT probíhá přenos pomocí TCP a používá návrhový vzor publisher – subscriber. Tedy existuje jeden centrální bod (MQTT broker), který se stará o výměnu zpráv. V Našem případě bude brokerem **Raspberry Pi**. Zprávy jsou tříděny do tzv. témat (topic) a zařízení buď publikuje v daném tématu (publish), to znamená, že posílá data brokeru, který je ukládá a distribuuje dalším zařízením, nebo je přihlášeno k odběru tématu či témat (subscribe)

a broker pak všechny zprávy s daným tématem posílá do zařízení. Jedno zařízení samozřejmě může najednou být v některých tématech publisher, v jiných subscriber.

Velikost zprávy je v aktuální verzi protokolu omezena, je to necelých 256 MB, ale naprostá většina zpráv je mnohem menší. (Protokol MQTT: komunikační standard pro IoT, 2016)

6.3 Vstupní panel

Zdrojový kód vstupního panelu u hlavního vchodu a u vchodu do kotelny jsou téměř totožné, proto zde popíši hlavní funkce pouze jednou. Jeho účelem bude umožnit zamčení domu stiskem klávesy, jakožto i jeho odemčení pomocí zadání 4místného číselného kódu. Současně s tím bude sloužit jako prvek k upozornění na vetřelce pomocí pasivního reproduktoru.

6.3.1 Popis zdrojového kódu

Na následujícím obrázku je vidět, že začátek kódu obsahuje makra, jež umožňují definovat, kolikrát je možné zadat pin při zamknutém domě, než dojde k časové prodlevě, která je definovaná hned na následující řádku. Do tohoto zařízení jsou zahrnuty celkem 4 knihovny, které usnadňují práci s připojením k Wi-Fi síti, zobrazení dat na displeji, čtení zmáčknutých kláves a komunikaci pomocí MQTT protokolu.

```
#define POCET_POKUSU 3 // počet pokusů, než bude čekat na další možnost zadat kód
#define CEKANI 15      // délka čekání při špatně zadaných X pokusech

//<><><><><><><><><><><><><><><><><><><><><><><><><><><><>//
//===============KNIHOVNY====================//
//<><><><><><><><><><><><><><><><><><><><><><><><><><><><>//
#include <WiFi.h>     // Slouží k připojení k wifi síti
#include <LiquidCrystal_I2C.h> // Pro snadnou obsluhu LCD displeje
#include <Keypad.h> // Pro čtení kláves z 4x4 klávesnice
#include <PubSubClient.h> // Připojení k MQTT brokeru
//<><><><><><><><><><><><><><><><><><><><><><><><><><><><>//
```

Obrázek 17 Kód vstupního modulu – Knihovny/Makra

Následující smyčka slouží ke konfiguraci zařízení, kde je nejprve nadefinována rychlost pro sériovou komunikaci. Následně je definován pin a Inicializován LCD displej. Poté následuje konfigurace Wi-Fi, realizována voláním funkce, jenž nejprve nastaví Wi-Fi jako standardní zařízení (druhou možností je AP neboli „Access Point“ tedy přístupový bod). Následně je ve funkci definován název sítě a heslo, ke které se má ESP připojit. Poté proběhne kontrola, jestli došlo k připojení k Wi-Fi a pokud ne, tak se pokouší připojit tak dlouho, dokud se nepřipojí. Současně je v tomto cyklu na displej vypsáno varovné hlášení s chybným připojením k Wi-Fi síti. Následuje funkce pro nastavení MQTT klienta, která definuje adresu brokeru a komunikační port. Poslední instrukce se stará o nastavení odběru zpráv od brokeru.

```
void setup() {
  Serial.begin(115200); // Nastavení rychlosti sériové komunikace
  pinMode(REPRODUCTOR,OUTPUT);
  digitalWrite(REPRODUCTOR,LOW);
  lcd.init();           // Inicializace LCD displeje
  lcd.backlight();      // Rozsvícení podsvícení displeje

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("NACITAM WIFI");
  wifiSetup();          // Připojení k wifi síti

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("NACITAM KOMUNIKACI");
  MQTTclient.setServer(MQTTserver, MQTTport); // Připojení k MQTT
  MQTTclient.setCallback(callback);   // Nastavení subscribe (odběru zpráv)
}
```

Obrázek 18 Kód vstupního modulu – Setup smyčka

Hlavní smyčka programu nejprve zavolá funkci, která ověří, jestli je ESP stále připojeno k Wi-Fi síti a pokud ne, tak jej zkouší 10krát připojit. Pokud se to ani poté nepovede, dojde k restartování zařízení. Následně je zavolána funkce, která ověří připojení k MQTT brokeru a pokud je připojen, vyčte příchozí zprávy od brokeru. Mezi příchozí zprávy patří stav, jestli je dům zamknut, v tom případě je tato skutečnost zobrazena na displeji, a v případě že při zamčeném stavu dojde k zaznamenání vetřelce z jedné z kamerových modulu, pak je spuštěn alarm v podobě pasivního reproduktoru, jenž vydává konstantní tón o určité frekvenci.
Poté následuje část, která zajišťuje logickou funkci tohoto zařízení. Nejprve je ověřeno, jestli uživatel stiskl tlačítko, pokud tak nedojde během 20 sekund, je displej automaticky zhasnut a rozsvícen je až v případě, že je stisknuto některé z tlačítek. Tato funkce je nezbytná z důvodu, aby displej nenarušoval kolemjdoucí svým nepřetržitým podsvícením.

Následně je ověřeno, jestli už uživatel nezadal třikrát špatný pin, jestliže ano, tak je následně ověřeno, jestli od poslední kontroly uplynula jedna sekunda. Pokud ano, je na druhý řádek zapsán aktuální stav, jak dlouho bude ještě uživatel čekat, než bude moci znovu zadat kód.

Pokud tedy uživatel nezadal třikrát špatný pin a bylo stisknuto tlačítko na klávesnici, ověřím, jaký je stav domu. Pokud je odemknut a byla stisknuta jakákoliv klávesa, pak dojde k zamknutí domu. Pokud byl však dům zamknut, pak načítám znaky do zásobníku. Ve chvíli, kdy je zadán 4. znak, dojde ke kontrole s PIN kódem domu. Pokud je kontrola v pořádku, pak dojde k odemknutí domu. Jestliže je však chybná, dojde k odečtení jednoho ze tří možných pokusů k zadání kódu do klávesnice. PIN kód je uložen v databázi, která se nachází na řídicí jednotce. Při spuštění zařízení vyšle zprávu s dotazem o aktualizaci na řídicí jednotku, ta obratem vyšle po zabezpečené komunikaci, jaký je stav domu a aktuální PIN kód.

```
void loop() {
  WifiCheck();
  mqttRun();
  char customKey = customKeypad.getKey(); // Čtení stisknuté klávesy
  if (customKey && !backligth) // funkce pro automaticke zhasinani
  {
    lcd.backlight();      // Rozsvícení podsvícení displeje
    backligth = true;
    lastClick = millis();
  }
  else if (backligth  && (millis() - lastClick) > 20000) // Po 20sek zhasni klavesnici
  {
    lcd.noBacklight();      // Rozsvícení podsvícení displeje
    backligth = false;
  }
  else
  {
  if (wrongPin >= POCET_POKUSU) { // Pokud byl pin X krát zadán špatně
    current_millis = millis();
    if (current_millis - last_capture_millis > 1000) // Pokud uběhla 1 sekunda
    {
      last_capture_millis = millis();
      if (newTry > 0)
      {
        lcd.setCursor(0, 1);lcd.print("                ");
        lcd.setCursor(0, 1);lcd.print("ODPOCET :");
        lcd.print(newTry);lcd.print("s");
        newTry--;
      }
      else
      {
        lcd.setCursor(0, 1);lcd.print("                ");
        wrongPin = 0; newTry = CEKANI;
      }
    }
  }
  else if (customKey) // Pokud byla stisknutá klávesa
  {
    lastClick = millis();
    pinInput[increment] = customKey;
    if (locked)   // Pokud je zamčen dům
    {
      if (increment == 0) // Slouží k vymazání znaků na druhém řádku displeje
      {
        lcd.setCursor(0, 1);lcd.print("                ");
      }
      lcd.setCursor(increment + 6, 1);
      lcd.print("*");
      increment++;
      if (increment > 3)  // Pokud byly zadány 4 znaky tak je ověř jestli jsou správné
      {
        for (increment = 0; increment <= 3; increment++)
        {
          if (pin[increment] != pinInput[increment]) // Když je některý znak špatný
          {
            lcd.setCursor(2, 1);
            lcd.print("SPATNE HESLO");
            wrongPin++;
            break;
          }
          else if (increment == 3) // Když nebyl ani jeden špatný pak odemkni dům
          {
            lcd.setCursor(4, 1);
            lcd.print("ODEMYKAM");
            wrongPin = 0; newTry = CEKANI;
            mqttRun();
            MQTTclient.publish(Publish, "0", MQTTpubQos); // Vyšli zprávu o odemknutí domu
          }
        }
        increment = 0;
      }
    }
```

Obrázek 19 Kód vstupního modulu – Hlavní smyčka 1/2

```
else// Když není dům zamknutý a je stiknutá klávesa tak dům zamkni
    {
      for(int LeaveHouse=600;LeaveHouse>=0;LeaveHouse--)
      {
        if ((LeaveHouse%10) == 0)
        {
        lcd.setCursor(0, 1);
        lcd.print("                ");
        lcd.setCursor(4, 1);
        lcd.print("ZAMYKAM: ");
        lcd.print(LeaveHouse/10);
        lcd.print("s");
        }
        delay(100); // Prodleva k opusteni domu
        customKey = customKeypad.getKey(); // Čtení stisknuté klávesy
        if (customKey) // Pokud byla stisknutá klávesa
          {
             lcd.setCursor(0, 1);
             lcd.print("                ");
             lcd.setCursor(2, 1);
             lcd.print("RUSIM AKCI");
             LeaveHouse=-1;
             delay(1000); // Prodleva k opusteni domu
             lcd.setCursor(0, 1);
             lcd.print("                ");
          }
      }
      lastClick = millis();
      if(!customKey)
      {
        Serial.println("Posilam zpravu o zamknuti domu");
        mqttRun();
        MQTTclient.publish(Publish, "1", MQTTpubQos);  // Vyšli zprávu o zamknutí domu
      }
    }
  }
  }
}
```

Obrázek 20 Kód vstupního modulu – Hlavní smyčka 2/2

6.4 Kamerový modul

Jeho primárním cílem je zaznamenat pohyb v určité oblasti, pokud byl předem modul aktivován. Pohyb je zaznamenán pomocí PIR senzoru, a následně jsou s vteřinovým intervalem pořizovány fotografie, dokud PIR senzor zaznamenává pohyb.

Sekundárním cílem je monitorování teploty a odesílání těchto dat řídicí jednotce.

6.4.1 Popis zdrojového kódu

Kód obsahuje makra, jež umožňují definovat, jak dlouhá je prodleva mezi jednotlivými pořízenými snímky, stejně jako prodleva mezi záznamem teploty. Další makra slouží definování pinů, na nichž je připojen PIR modul a teplotní čidlo, stejně tak přesnost, s jakou je měřena hodnota na čidlu. Dále jsou zde definovány katody RGB led. Do tohoto zařízení je zahrnuto celkem 6 knihoven, které zajišťují odeslání pořízeného snímku serveru, načtení kamery, připojením k Wi-Fi síti, komunikaci pomocí MQTT protokolu a připojení a práci s teplotním čidlem.

```
void setup()
{
  Serial.begin(115200); // Nastavení rychlosti sériové komunikace
  pinMode(LedR, OUTPUT);
  pinMode(LedG, OUTPUT);
  pinMode(LedB, OUTPUT);
  Signalization('g');
  pinMode(PIR, INPUT);  // Nastaveni PIR pinu jako vstup
  pinMode(ONE_WIRE_BUS, INPUT); // Nastaveni pinu teplotního čidla jako vstup

  wifiSetup(); // Připojení k wifi síti

  MQTTclient.setServer(MQTTserver, MQTTport); // Připojení k MQTT
  MQTTclient.setCallback(callback); // Nastavení subscribe (odběru zpráv)

  camera_config_t config; // Nastavení kamery
  esp_err_t err = esp_camera_init(&config); // Pokud je inicializace kamery chybná
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    ESP.restart();
    return;
  }

  sensors.begin(); // Inicializace tepltoního čidla
}
```

Obrázek 21 Kód kamerového modulu – Setup smyčka

Smyčka *Setup* je téměř identická s tou v kódu vstupních dveří, ovšem zde není displej, klávesnice ani reproduktor, namísto toho je konfigurována a inicializována kamera a teplotní čidlo spolu s PIR modulem a RGB led.

```
//<><><><><><><><><><><><><><><><><><><><><><><><>//
//===============KNIHOVNY=====================//
//<><><><><><><><><><><><><><><><><><><><><><><><>//
#include "esp_http_client.h" // Pro odeslání snímku na server
#include "esp_camera.h"  // Pro konfiguraci a obsluho kamery
#include <WiFi.h> // Slouží k připojení k wifi síti
#include <PubSubClient.h> // Připojení k MQTT brokeru
#include <OneWire.h>   // Slouží pro komunikaci s teplotním čidlem
#include <DallasTemperature.h> // Umožňuje číst hodnotu s čídla

#define capture_interval 1000 // Mikrosekundy mezi pořízením snímku (1 sekunda)
#define temp_interval 300000 // Mikrosekundy mezi zaznamenáním teploty (5min)

#define PIR 15 // PIR připojen na pinu
#define ONE_WIRE_BUS 14 // Teplotní čidlo připojeno na pinu
#define TEMPERATURE_PRECISION 12 // Přesnost měření teploty
#define LedR 4 // červená katoda RGB LED
#define LedG 2 // zelená katoda RGB LED
#define LedB 13 // modrá katoda RGB LED
/*
 * žlutá                      - ztráta komunikace s wifi
 * fialová                    - ztráta komunikace s mqtt brokerem
 * červena                    - chyba kamerového modulu
 * zelené probliknuti         - startuji
 */
```

Obrázek 22 Kód kamerového modulu – Knihovny/Makra

V hlavní smyčce programu tohoto modulu jsou nejprve volány funkce pro ověření připojení k Wi-Fi síti a kontrola na připojení k MQTT brokeru. Tyto funkce jsou totožné s těmi v předchozím modulu. Následně je kontrolováno, jestli od poslední kontroly neuběhla doba určená k měření teploty (nastaveno na 5 minut). Pokud tomu tak je, tak je zavolána funkce, jejichž úkolem je načíst teplotu z čidla, tu následně zpracovat do stringového řetězce, který je poslán na broker. Další podmínka kontroluje, jestli nepřišla z brokeru zpráva o zamčeném domě. Jestliže se tak stalo, pak je kontrolován pohyb v oblasti snímáním hodnoty z PIR senzoru. Pokud byl zaznamenán pohyb, je poslána na broker zpráva o vniknutí do domu. Současně s tím je kontrolováno, jestli od poslední smyčky neuběhla doba potřebná pro vyfocení dalšího snímku obrazovky. Pokud ano, je zavolána funkce, která má za úkol pořídit snímek pomocí připojené kamery. Následně je tento snímek odeslán pomocí http metody POST, na stránku Raspberry Pi serveru, kde je snímek pomocí php skriptu přijat a poté uložen do adresáře.

```
void loop()
{
  WifiCheck();

  mqttRun();

  temp_current_millis = millis();
  if (temp_current_millis - temp_last_capture_millis > temp_interval) // Pokud
uplynula doba pro další měření teploty
  {
    temp_last_capture_millis = millis();
    Teplota();
  }

  if (locked) // Pokud je dům zamknut
  {
    bool PIRKO = digitalRead(PIR);
    if (PIRKO) // Pokud je zaznamenán pohyb
    {
      if (!last_PIR)
      {
        Serial.println("Public Intruder");
        MQTTclient.publish(PublishIntruder, MQTTid, MQTTpubQos); // Publikuj zprávu o
zaznamenaném pohybu
      }
      current_millis = millis();
      if (current_millis - last_capture_millis > capture_interval)  // Pokud uběhla 1
sekunda
      {
        last_capture_millis = millis();
        take_send_photo(); // Vyfoť a pošli snímek
      }
    }
    else
    {
    last_PIR = PIRKO;
    }
  }
}
```

Obrázek 23 Kód kamerového modulu – Hlavní smyčka

6.5 Řídicí jednotka

Účely této jednotky jsou následující. Zprostředkovat komunikaci pomocí MQTT protokolu. Ukládat příchozí data do databáze a umožnit k nim přístup. Zajistit HMI interface.

6.5.1 Databáze

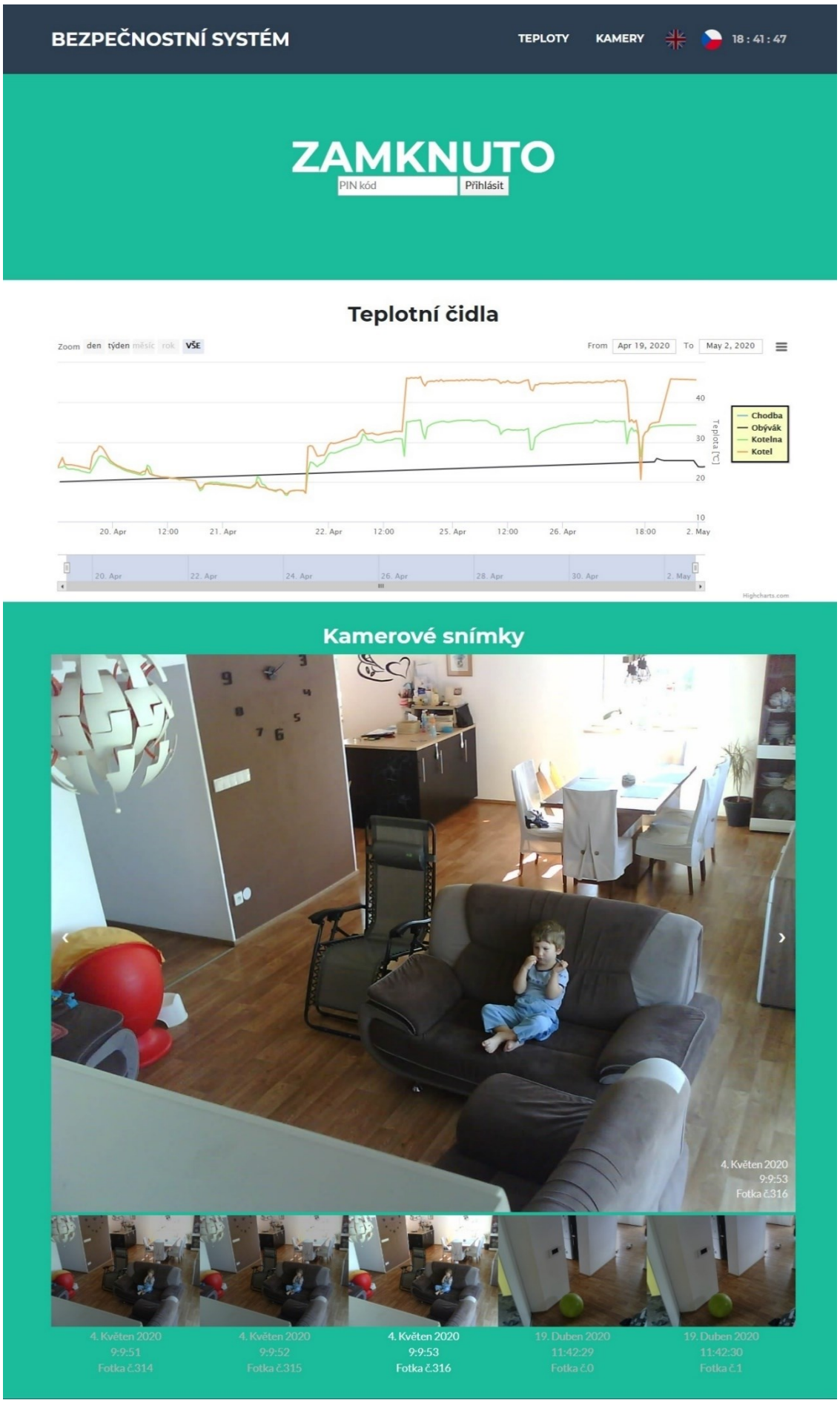
Pro ukládání dat využívám databáze MariaDB, jenž je relační databáze, která je komunitou vyvíjenou nástupnickou větví MySQL. V databázi jsou vytvoření 4 uživatelé s rozdílnými právy. To je z důvodu, abych co možná nejvíce zabezpečil přístup k informacím. Hlavním uživatelem je *root*, tento uživatel má neomezená práva, co se databáze týče. Uživatel *Admin* má neomezená práva vztahující se na databázi *esp_db*, ve které jsou uloženy všechny tzv. *TABLE*. Posledními uživateli jsou pak *Write*, který může do jednotlivých *TABLE* zapisovat a uživatel *Read*, který může data pouze číst. Jak jsem jíž zmínil, je zde hlavní databáze *esp_db*, ve které jsou následující *TABLE*. Pro uložení teplot slouží *ESP_*, namísto _ je číslovka označující dané zařízení, v nich nalezneme sloupce času a hodnot teploty s desetinnou čárkou. Dalším je *HOUSE*, ve kterém se nachází rovněž sloupec času a druhým je stav zamknutí/odemknutí domu vyjádřen pomocí booleovy logiky. Následující je *INTRUDER*, ve kterém je rovněž uchován čas a dalším sloupcem je pak stav (0/1). Přičemž 1 můžou zapsat pouze kamerové moduly a tento stav je tam až do doby, než je dům odemknut. Dojde-li tedy k vniknutí vetřelce, pak jsou kamerové moduly a alarm aktivní, dokud nedojde k zadání pin kódu, který odemkne systém. Další je pak tabulka, která nese název *PHONES*. V této tabulce jsou uloženy telefonní čísla a názvy uživatelů, kterým tyto telefonní čísla patří. Další dva sloupce této tabulky označují práva jednotlivých telefonních čísel, jeden sloupec označuje právo pomocí SMS odemknout či zamknout dům, zatímco druhý sloupec označuje manipulaci s bránou. Posledním je *LOGIN*, ve kterém je uložen 4místný PIN pro odemknutí domu.

6.5.2 Web server

Jak již bylo zmíněno, na Raspberry běží web server, který je zprostředkován službou Apache 2. Na této službě běží řada php skriptů doplněných o HTML strukturu. O dynamiku stránek se pak stará JavaScript spolu s jQuery. Vše je designově sjednoceno a zformátováno pomocí css stylu. Kompletní kódy jsou dostupné v příloze, já zde nastíním funkce jednotlivých souborů.

Soubor s názvem *NactiFotkuESP_x.php* slouží jako výchozí adresa pro kamerové moduly, do něj jsou odesílány fotky z jednotlivých zařízení.

Dalším funkčním souborem je *MySQL.php*, v něm jsou uloženy všechny funkce, které slouží jako prostředník v případě, že chci data z databáze číst, nebo do ní zapisovat.

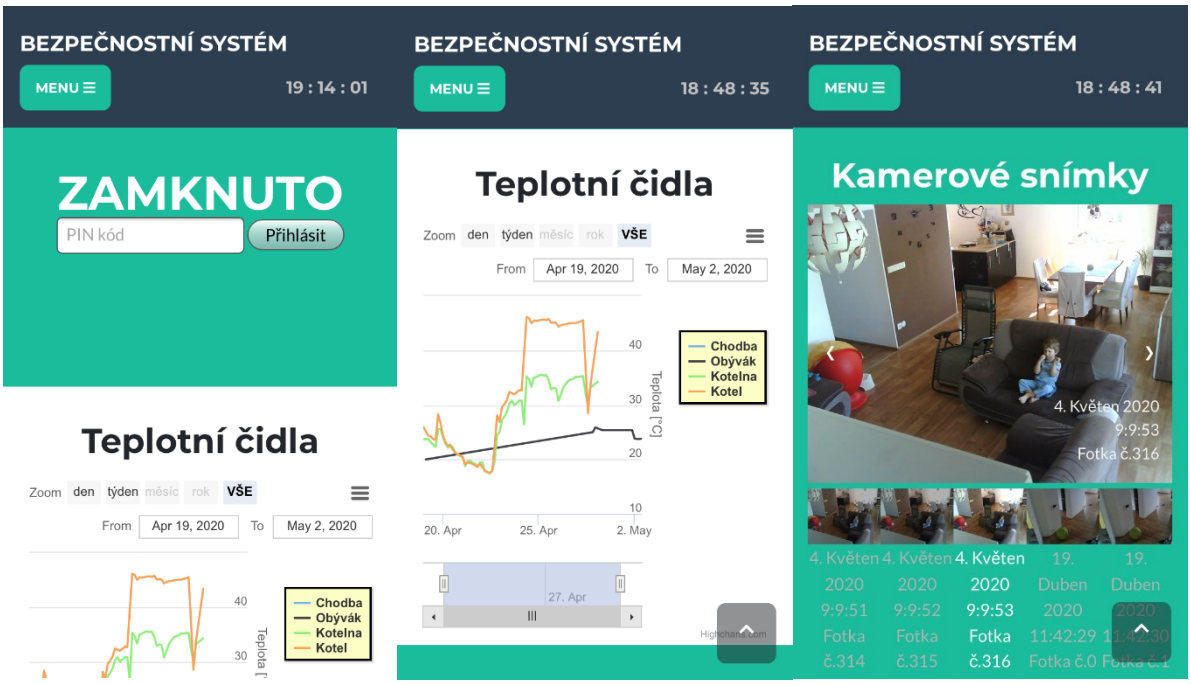


Obrázek 24 Web server

Výše uvedený web server je rozdělen do tří částí. V první se nachází možnost přihlásit se pomocí zadání PIN kódu. Po úspěšném přihlášení je možné vzdáleně dům odemknout či zamknout.

V další sekci se pak nachází HighCharts graf, který přehledně zobrazuje průběh teplot v čase naměřených na jednotlivých zařízeních. V grafu je možné vybrat určité zařízení, od kterého chceme vidět průběh teplot. Stejně tak je možné si zvolit, v jakém časovém rozsahu chceme teplotu zobrazit. Nachází se zde i doplňující možnosti, ve kterých se dají grafy exportovat ať už do obrázku, nebo například do excel sešitu, či podobných souborů.

V poslední sekci s názvem kamery nalezneme všechny pořízené kamery ze všech připojených kamerových zařízení, které jsou k sestavě připojeny. Tyto snímky jsou seřazeny podle data, které je zobrazeno u každé fotografie a pro větší přehlednost je jim přiděleno i pořadové číslo. V zápatí stránky se nachází kontakt, který je zde uveden v případě, že nastane situace, která znemožní správné fungování systému. V případě potřeby je stránky možné přepnout i do anglického jazyka, pomocí zvolení anglické vlajky v záhlaví stránky. Rovněž jsou tyto stránky přizpůsobeny i pro mobilní zařízení, jakým je například mobilní telefon. Předpokládám totiž, že tyto stránky budou využívat převážně tyto zařízení.



Obrázek 25 Web server - mobilní zařízení

6.5.3 Python skripty

Tyto skripty slouží ke zprostředkování komunikace pro MQTT protokol, stejně tak k ukládání dat do databáze a komunikaci s GSM modulem. Na Raspberry se nachází tři python soubory, jeden z nich se chová jako MQTT klient. Poslouchá vše, co je na broker zasláno a na základě příchozí zprávy vykoná určitou akci. Druhý python skript pak slouží jako podpora pro první. V něm jsou funkce, které umožňují přístup do databáze. Třetí pak komunikuje po sériové lince s GSM modulem a zprostředkovává komunikaci s koncovým uživatelem.

Druhý python skript obsahuje funkce, které slouží k načtení hodnot z databáze, jako je například aktuální stav domu (zamknut/odemknut). Druhá pak k zapisování například hodnot do tabulek s teplotami, které přijdou jako zpráva od kamerových modulů prostřednictvím MQTT protokolu. Celé python skripty jsou uvedeny v příloze práce, já zde uvedu pouze klíčové funkce.

Níže zmíněné funkce pracují s MQTT komunikací. První z nich, *on_message*, je volána v případě, že je mqtt broker vyšle nějakou zprávu. Ve funkci je pak zpráva filtrována pomocí topicu, kde se zajímám o to, jaký název zpráva nese. V případě, že se jedná o topic s názvem *Refresh*, tak vím, že se právě do komunikace připojilo nové zařízení a žádá o aktualizaci údajů. Na tento povel je tedy zaslána zpráva, která zavolá funkci, jenž vyčte z databáze aktuální pin kód a stav domu. Tyto zprávy odešle po zašifrované komunikaci.

Druhá funkce se pak stará o změnu stavu domu (odemknutí/zamknutí). V případě, že k ní nastane, dojde po 5 vteřinách k opakovanému vyslání zprávy, aby bylo 100 % jisté, že všechny připojené zařízení zprávu obdrží. Se stejnou frekvencí jsou pak vysílány zprávy i o stavu vetřelce (tedy jestli byl jedním z kamerových zařízení zaznamenán pohyb), pokud je dům v zamknutém stavu.

```
def on_message(client, userdata, message):
    global last_LockState
    if ("Refresh" in message.topic):
        #print("Pozadavek na Refresh")
        client.publish('Read/ESP/Lock', payload=last_LockState, qos=1,
retain=False)
        time.sleep(1)
        client.publish('Read/ESP/Pin', payload=ReadFromDB("Read/ESP/Pin"),
qos=1, retain=False)
        time.sleep(1)
        client.publish('Read/ESP/Intruder',
payload=ReadFromDB("Read/ESP/Intruder"), qos=1, retain=False)
    elif ("Write" in message.topic):
        SaveToDB(message.topic, message.payload.decode('utf-8'))
        print("Topic: " + message.topic + "  Data: " +
message.payload.decode('utf-8'))
        client.publish('Read/ESP/Lock', payload=last_LockState, qos=1,
retain=False)
        time.sleep(1)
        client.publish('Read/ESP/Intruder',
payload=ReadFromDB("Read/ESP/Intruder"), qos=1, retain=False)

def publish_state():
    global last_LockState
    while client.connected_flag:
        time.sleep(5)
        if ReadFromDB('Read/ESP/Lock') != last_LockState:
            last_LockState = not last_LockState
            client.publish('Read/ESP/Lock', payload=last_LockState, qos=1,
retain=False)
            time.sleep(1)
        if last_LockState:
            client.publish('Read/ESP/Intruder',
payload=ReadFromDB("Read/ESP/Intruder"), qos=1, retain=False)
```

Obrázek 26 Python skript pro komunikaci s MQTT

Skript, který zprostředkovává komunikaci s GSM modulem, nejprve tento modul nastaví. Poté přejde do nekonečné smyčky, ve které kontroluje, jestli nedošlo k vniknutí vetřelce do domu. Pokud ano, pak je zaslána SMS zpráva na všechny telefonní čísla, která jsou v databázi registrovaná. Pokud dojde k vytáčení telefonního čísla, je číslo nejprve ověřeno, jestli je registrováno v databázi a má patřičná práva k otevření brány. V případě, že ano, dojde ke změně stavu na obsazeno (aby nespadl do hlasové schránky). Poté dojde k sepnutí kontaktu na reléovém modulu, který je k Raspberry Pi připojen a tím dojde k otevření brány (tato funkce byla přidána na základě žádosti majitele, kde je systém instalován). Pomocí zaslání SMS „zamknout“ nebo „odemknout“ je možné manipulovat se zabezpečovacím systém domu. Samozřejmě je i při zaslání SMS nejprve provedena kontrola, jestli je číslo, z kterého byla zaslána SMS , registrováno v databázi a má uživatel patřičná povolení. Následně je na toto telefonní číslo odeslána SMS zpráva. Tato zpráva obsahuje aktuální čas, stav domu (odemknuto/zamknuto), poslední naměřené teploty ze všech čidel a v poslední řadě pak

zbývající kredit na SIM kartě (aby uživatel věděl, kdy je nutné Twist kartu dobít). Na žádost majitele, byla přidána funkce, která kontroluje teploty kotle, jež jsou zaslány z jednoho z modulu, a pokud dojde k překročení hranice 80 °C, je majitel informován SMS zprávou.

```
reply = bytes.decode(ser.read(ser.inWaiting())) # čtení dat po sériové
lince
if reply !="":
    print (reply)
    if "RING" in reply:  # if calling
        phoneNumber = reply[reply.index("+CLIP: ") +
8:reply.index("+CLIP: ") + 20]
        if phoneNumber in PhoneTable[:][0] and 1 in PhoneTable[:][2]: #
pokud je telefonní číslo v databázi
            GPIO.output(17, GPIO.LOW) # relé sepnuto
            time.sleep(2)
            GPIO.output(17, GPIO.HIGH)# relé rozepnuto
        ser.write(str.encode('ATH\r'))  # změna stavu na obsazeno

    if '+CMT: ' in reply:
        phoneNumber = reply[reply.index("+CMT: ") + 8:reply.index("+CMT:
") + 20]
        print (phoneNumber)
        if phoneNumber in PhoneTable[:][0] and 1 in PhoneTable[:][1]: #
pokud je telefonní číslo v databázi
            if "zamknout" in reply.lower() or "zamkni" in reply.lower():
# pokud obsahuje toto klíčové slovo
                print("Zamykam DUM")
                ChangeState("true")
                time.sleep(1)
                SendSMS(phoneNumber)
            elif "odemknout" in reply.lower() or "odemkni" in
reply.lower():
                print("Odemykam DUM")
                ChangeState("false")
                time.sleep(1)
                SendSMS(phoneNumber)
            else:
                SendSMS(phoneNumber)
    bytes.decode(ser.read(ser.inWaiting()))
    ser.flushInput()
    ser.flushOutput()
time.sleep(1)
```

Obrázek 27 Hlavní smyčka python skriptu pro komunikaci s GSM modemem

7 Návrh krycích a montážních pouzder pro jednotlivá zařízení

Aby bylo možné zařízení implementovat do provozu, tedy umístit je do domu, je nutný předpoklad, aby měly tyto zařízení mechanickou ochranu a současně umožňovaly co možná nejsnadnější montáž. Z toho důvodu jsem se rozhodl vyrobit na 3D tiskárně pouzdra pro kamerové moduly a pro vstupní panel. Na Raspberry Pi je k dispozici nepřeberné množství pouzder, takže není nutné navrhovat vlastní. Návrh všech pouzder jsem prováděl v programu Autocad Inventor a k tisku jsem využil 3D tiskárnu Ender 3 PRO.

7.1 Vstupní panel s membránovou klávesnicí

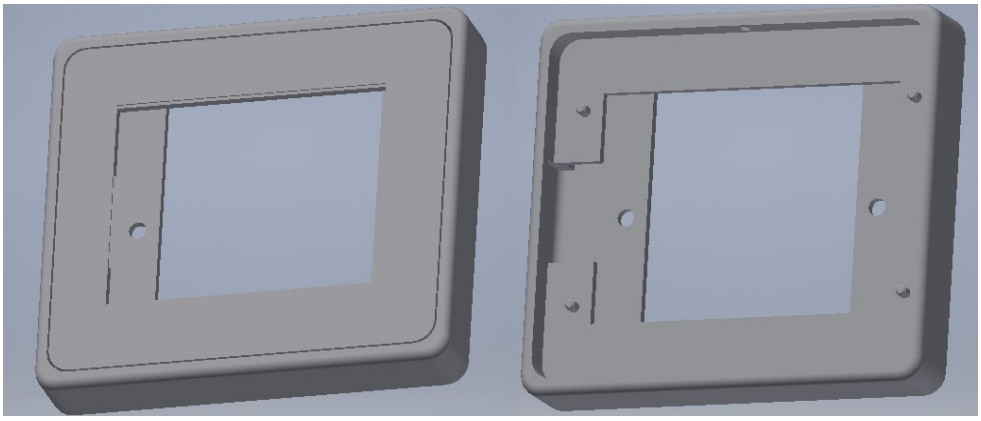
Do vstupního panelu je nutné uložit ESP32, LCD displej, klávesnici, reproduktor a DC-DC měnič napětí, který bude snižovat rozvodové napětí 24V DC na potřebných 5V DC.

Pouzdro je určeno k montáži na zeď v prostorách vstupních dveří, s výškou vyhovující uživateli.

Obrázek 28 Návrh pouzdra pro vstupní panel

7.2 Vstupní panel s dotykovým displejem

Do tohoto zařízení je umístěn dotykový displej, mikrokontrolér ESP32 a DC-DC měnič napětí.

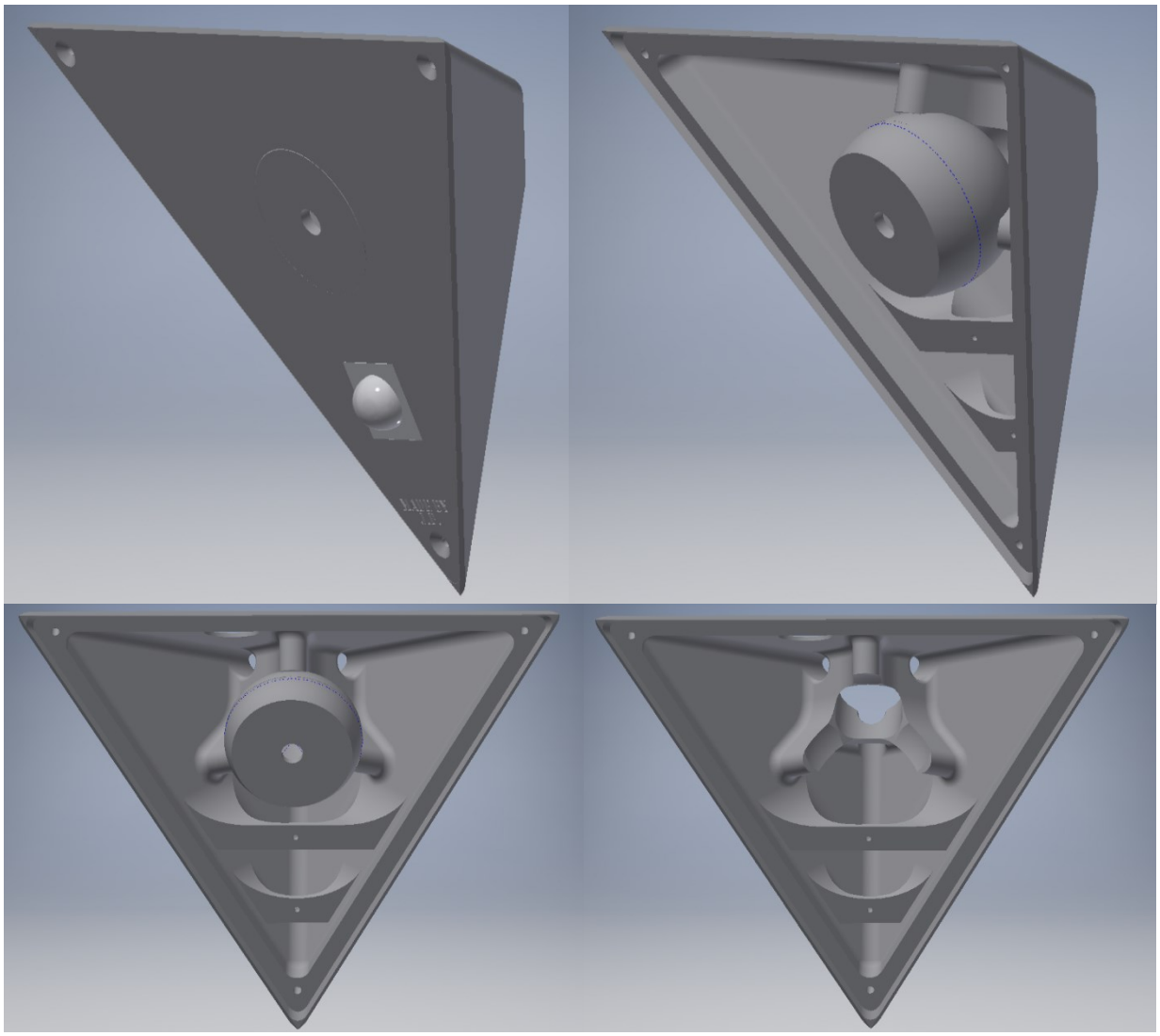


Obrázek 29 Dotykový vstupní panel

Kamerový modul

Kamerový modul bude čítat následující prvky: ESP32-CAM, teplotní čidlo, PIR senzor, RGB led, IR led, NPN tranzistor, zdroj pro konstantní proud, DC-DC měnič, anténu pro Wi-Fi.

Pouzdro je určeno k montáži v rohu místnosti tak, aby co nejvíce splynula s okolními stěnami a nijak nenarušovala celistvost místnosti. Kamera je umístěna v otočné kouli, aby bylo možné nastavit, jaká oblast místnosti má být zaznamenávána.



Obrázek 30 Návrh pouzdra pro kamerový modul

8 Instalace a testování zabezpečovacího systému

Do domu byly nainstalovány dva kamerové moduly. Dva vstupní panely, z nichž jeden je s dotykovým displejem a druhý s membránovou klávesnicí (snazší manipulace v prašné místnosti). Jedním modulem pro sběr teplot v kotelně a přímo z kotle. A jednou řídicí jednotkou coby Raspberry Pi Zero W.

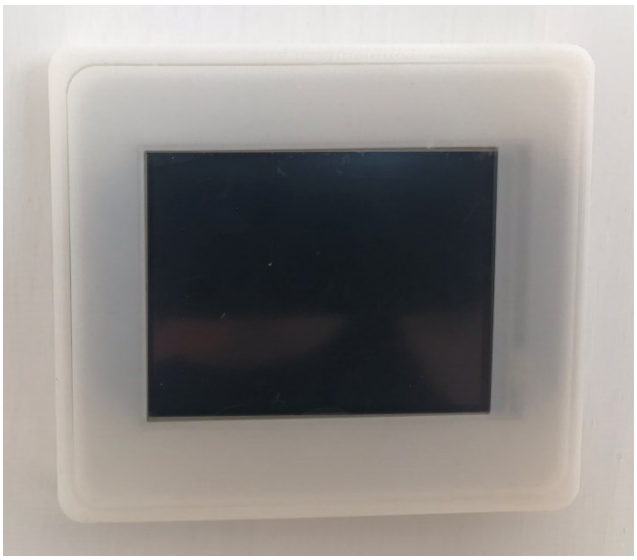
Všechny komponenty zařízení jsou napájeny z 12V zdroje se stejnosměrným napětím, aby při i dlouhých trasách bylo u jednotlivých zařízení napětí nejméně 5 V.

Na obrázku níže je jeden ze dvou kamerových modulů, který byl právě aktivován a signalizuje svojí inicializaci pomocí zelené barvy.

Obrázek 31 Montáž kamerového modulu

Na dalších snímcích je možné vidět vstupní panely. Vlevo se nachází s membránovou klávesnici, jenž je umístěn v kotelně a druhý se pak nachází u hlavního vchodu.

Obrázek 32 Vstupní panel



Obrázek 33 Dotykový vstupní panel

Na následujícím snímku je zobrazeno zařízení, které je umístěno u kotle a jehož cílem je monitorování jeho teploty a teploty okolí.

Obrázek 34 Zařízení pro sběr teplot

V rozvaděčové skříni jsou nainstalovány Raspberry Pi Zero W, spolu s GSM modemem a reléovým modulem.

Po zapojení všech modulů došlo k otestování jednotlivých zařízení, došlo k doladění některých částí kódu tak, aby byla zajištěna co možná největší spolehlivost. Po delším provozu byly zjištěny občasné výpadky Wi-Fi u některých zařízení. Za tímto problémem však nestála porucha v sestavě, nýbrž v pokrytí domu Wi-Fi signálem a stabilitou modemu. Problém jsem vyřešil instalací nového modemu, a lepší pokrytí Wi-Fi signálu instalací opakovače.

9 Závěr

Blíže jsem se seznámil s mikroprocesorem ESP32, způsoby, jakým jej lze programovat a možnostmi jeho využití. Začínal jsem s jednoduchými aplikacemi, které využívají tento mikroprocesor. Takto jsem dále postupoval, až jsem se dopracoval k rozšíření o kamerový modul. Nabité zkušenosti mě zavedly k aplikaci kamery na mnou navrženou úlohu. To mě dále vedlo k využití této sestavy v oblasti zabezpečovací techniky domu.

Navrhl jsem tedy soustavu kamer, která byla nainstalována na rodinném domě. Aby však kamery pouze nenatáčely a jednalo se o smysluplný systém, využil jsem jejich možnost připojit se k Wi-Fi síti. Sestavu jsem dále rozšířil o vstupní panely u hlavního vchodu a u vchodu do kotelny. Bylo nutné vytvořit nadřazené zařízení, které bude sloužit jako úložiště dat a řídicí prvek celé sestavy. Bohužel však nemá k této úloze ESP32 dostatečnou kapacitu. Jako řídicí jednotku jsem tedy vybral Raspberry Pi, jejíž funkcí je předávání komunikace pro všechny ostatní prvky v sestavě. Současně s tím slouží jako úložiště fotografií pořízených z kamerových modulů. Rovněž na něm běží i web server, který umožňuje přístup k datům na této jednotce.

V klidovém stavu jsou kamerové moduly neaktivní (nepořizují kamerové snímky). Pokud však dojde k zamknutí domu, který je možné zamknout pomocí dvou vstupních panelů nebo pomocí webserveru či pomocí zaslání SMS příkazu, pak dojde po určité časové prodlevě dostačují pro opuštění domu k aktivaci kamerových modulů. Ty kontrolují, jestli nedošlo k zaznamenání pohybu na PIR senzorech. Pokud tato situace nastane, pak je tato informace zaslána na řídicí jednotku a rozeslána na všechny kamerové moduly. Ihned nato jsou ze všech kamerových modulů pořizovány snímky s vteřinovým intervalem. Rovněž dojde k aktivaci alarmu, který je umístěn v jednom vstupním panelu a z řídicí jednotky je prostřednictvím GSM modemu odeslána SMS zpráva na určitá telefonní čísla o tom, že došlo k vniknutí do domu.

Snímky pořízené z kamerových modulů jsou prostřednictvím http post odeslány na řídicí jednotku, kde dojde k jejich uložení.

Aby byla zařízení využívána i v době kdy nejsou monitorovány prostory domu, jsou kamerové moduly osazeny i teplotními čidly. Tyto teploty jsou zpracovány a odeslány na řídicí jednotku, kde dojde k jejich uložení do databáze. Pro smysluplný sběr dat je sestava ještě obohacena o jedno zařízení, které načítá teploty v kotelně a přímo z kotle. Tím je umožněno sledovat závislost mezi teplotami na kotli a v jednotlivých místnostech. Díky tomu by v budoucnu mohlo dojít k efektivnějšímu vytápění rodinného domu.

O snadný a přehledný přístup ke všem datům, jež jsou uložena na řídicí jednotce, se stará webserver, který rovněž běží na této jednotce. Na webserveru je zobrazen aktuální stav domu

a možnost manipulace s ním po zadání PIN kódu. Níže se pak nachází graf, který přehledně vykresluje teploty naměřené na jednotlivých zařízeních v závislosti na čase. Je zde možnost filtrování určitého zařízení, ze kterého byly data naměřena, stejně jako možnost zobrazení v určitém časovém intervalu. Poslední sekcí webové stránky je galerie fotografií, které byly pořízeny na kamerových modulech a jsou seřazeny od nejnovější s tím, že u každé fotografie je uveden i přesný čas, kdy byla pořízena.

Jak jsem již zmínil, tak na řídicí jednotce je připojen GSM modem, ten umožňuje vzdálenou manipulaci s domem prostřednictvím SMS zprávy, ovšem pouze pro vybraná telefonní čísla. Pomocí zaslaní příkazu „odemknout“ nebo „zamknout“, dojde k vykonání této akce. Zpětně je pak uživatel informován o provedení pomocí SMS zprávy, která obsahuje aktuální stav domu (odemknut/zamknut). Dále pak poslední naměřené teploty ze všech připojených čidel a zůstatek kreditu na SIM kartě. Pokud dojde k zaslaní jakékoliv jiné SMS zprávy, opět pouze z vybraných čísel, je uživateli zaslána SMS se stejným obsahem jako při zaslání příkazu. Při prozvonění toho telefonního čísla dojde k sepnutí relé, které je připojeno k bráně pro vjezd na pozemek domu.

Dalšími kroky, jak sestavu vylepšit by bylo vyřešení problému pořizování kamerových snímků za zhoršených viditelnostních podmínek či v úplné tmě. Tento problém by bylo možné vyřešit odstraněním IR filtru, který se nachází na kamerových čočkách a nasvícením prostor přidáním například IR led, jejichž světlo by na sebe neupoutávalo pozornost, neboť není lidským okem viditelné. Dalším možným rozšířením by bylo přidání zařízení, které by zajišťovaly stabilnější regulaci teplot v domě, které jsou sbírany z téměř všech zařízení v sestavě. Realizace takového zařízení by se odvíjela od způsobu vytápění v domě.

Poděkování

Rád bych poděkoval doc. Ing. Marku Babiuchovi, Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování mé diplomové práce.

10 Bibliografie

2,8" USART displej od společnosti NEXTION [online], b.r. In: . [cit. 2020-01-17]. Dostupné z: https://docplayer.cz/docs-images/92/109654505/images/34-0.jpg

Ai-Thinker ESP32-CAM, b.r. In: *Laskarduino.cz* [online]. [cit. 2020-01-02]. Dostupné z: https://cdn.myshoptet.com/usr/www.laskarduino.cz/user/shop/big/2924-4_ai-thinker-esp32-cam-2-4ghz-wifi-bluetooth-modul.jpg

Alarm [online], b.r. In: . [cit. 2020-03-19]. Dostupné z: https://images-na.ssl-images-amazon.com/images/I/51qevxVXAnL._AC_SL1024_.jpg

Arduino IDE Tutorials, b.r. *Arduino* [online]. [cit. 2020-01-05]. Dostupné z: https://www.arduino.cc/en/Tutorial/HomePage

Arduino LCD displej, 2017. In: *Arduino Project* [online]. [cit. 2017-12-07]. Dostupné z: http://www.arduino-project.cz/520-large_default/arduino-lcd-displej-1602-radic-iici2c.jpg

ESP-32 Wroom [online], b.r. In: . [cit. 2020-01-08]. Dostupné z: https://arduino-shop.cz/photos/produkty/d/1/1581.jpg?m=1502871488

Espressif Systems [online], b.r. Shanghai: Espressif Systems [cit. 2020-01-05]. Dostupné z: http://espressif.com/en/

Function block diagram for Espressif's ESP32 series of Wi-Fi/Bluetooth chips., b.r. In: *Commons.wikimedia.org* [online]. Creative Commons [cit. 2020-01-02]. Dostupné z: www: https://commons.wikimedia.org/wiki/File:Espressif_ESP32_Chip_Function_Block_Diagram.svg#/media/File:Espressif_ESP32_Chip_Function_Block_Diagram.svg

GSM modul [online], b.r. In: . [cit. 2020-03-29]. Dostupné z: https://www.iotwebplanet.com/wp-content/uploads/2019/02/IOT-GA6-B-GSM-Module-600x600.jpg

KOLBAN, Neil, 2016. *Kolban's Book on the ESP32 & ESP8266* [online]. [cit. 2019-02-18].

KOLBAN, Neil, 2018. *Kolban's book on ESP32* [online]. [cit. 2019-02-18].

Maxim Integrated DS18B20: [katalogový list] [online], b.r. [cit. 2020-01-05]. Dostupné z: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

Membránová klávesnice [online], b.r. In: . [cit. 2020-01-08]. Dostupné z: https://arduino-shop.cz/photos/produkty/f/0/824.jpg?m=1550311640

PIR modul AM312 [online], b.r. In: . [cit. 2020-01-17]. Dostupné z: https://arduino-shop.cz/photos/produkty/d/7/7822.jpg?m=1565177586

PIR modul HC-SR501 [online], b.r. In: . [cit. 2020-01-17]. Dostupné z: https://www.gme.cz/data/product/1024_1024/pctdetail.775-042.1.jpg

Protokol MQTT: komunikační standard pro IoT, 2016. In: *Root.cz* [online]. [cit. 2019-02-18]. Dostupné z: https://www.root.cz/clanky/protokol-mqtt-komunikacni-standard-pro-iot/

Raspberry Pi 4 Model B [online], b.r. In: . [cit. 2020-01-17]. Dostupné z: https://rpishop.cz/6483-large_default/raspberry-pi-4-model-b-4gb-ram.jpg

Raspberry Pi Zero W [online], b.r. In: . [cit. 2020-01-08]. Dostupné z: https://rpishop.cz/2462-large_default/raspberry-pi-zero-w.jpg

Raspbian, b.r. *Raspberrypi* [online]. [cit. 2020-01-02]. Dostupné z: https://www.raspberrypi.org/downloads/raspbian/

STEHLÍK, Petr, 2016. ESP32 je tu. Co přinese nástupce ESP8266?. *Root.cz* [online]. ISSN 1212-8309. Dostupné také z: https://www.root.cz/clanky/esp32-je-tu-co-prinese-nastupce-esp8266/

VALÁŠEK, Michal, b.r. Raspberry Pi mění svět: Seznamte se s nejzajímavějším počítačem dneška. *Hospodářské Noviny IHNED* [online]. 1996-2020 Economia, a.s. [cit. 2020-01-02]. ISSN 1213-7693. Dostupné z: https://tech.ihned.cz/geekosfera/c1-65195330-raspberry-pi-meni-svet-seznamte-se-s-nejzajimavejsim-pocitacem-dneska

WAHER, Peter, 2015. *Learning Internet of Things*. Packt Publishing. ISBN 978-1783553532.

WOSTL, Marek, b.r. *Operační systémy pro Raspberry Pi 4 Model B* [online]. In: . [cit. 2020-01-02].

11 Seznam obrázků

Obrázek 1 ESP32 PINOUT (Stehlík, 2016) 9

Obrázek 2 ESP32 Funkční blokový diagram (Function block diagram for Espressif's ESP32 series of Wi-Fi/Bluetooth chips., b.r.) 10

Obrázek 3 Arduino IDE 12

Obrázek 4 Espressif IoT Development Framework 13

Obrázek 5 Architektura navržené úlohy 16

Obrázek 7 (Arduino LCD displej, 2017) 18

Obrázek 6 (ESP-32 Wroom, b.r.) 18

Obrázek 9 (Alarm, b.r.) 18

Obrázek 8 (Membránová klávesnice, b.r.) 18

Obrázek 10 (2,8” USART displej od společnosti NEXTION, b.r.) 19

Obrázek 11 (Ai-Thinker ESP32-CAM, b.r.) 20

Obrázek 12 (PIR modul HC-SR501, b.r.) 21

Obrázek 13 (PIR modul AM312, b.r.) 21

Obrázek 14 (Raspberry Pi 4 Model B, b.r.) 21

Obrázek 15 (Raspberry Pi Zero W, b.r.) 22

Obrázek 16 (GSM modul, b.r.) 22

Obrázek 17 Kód vstupního modulu – Knihovny/Makra 24

Obrázek 18 Kód vstupního modulu – Setup smyčka 25

Obrázek 19 Kód vstupního modulu – Hlavní smyčka 1/2 26

Obrázek 20 Kód vstupního modulu – Hlavní smyčka 2/2 27

Obrázek 21 Kód kamerového modulu – Setup smyčka 28

Obrázek 22 Kód kamerového modulu – Knihovny/Makra 28

Obrázek 23 Kód kamerového modulu – Hlavní smyčka 29

Obrázek 24 Web server 31

Obrázek 25 Web server - mobilní zařízení 32

Obrázek 26 Python skript pro komunikaci s MQTT 34

Obrázek 27 Hlavní smyčka python skriptu pro komunikaci s GSM modemem 35

Obrázek 28 Návrh pouzdra pro vstupní panel 36

Obrázek 29 Dotykový vstupní panel 36

Obrázek 30 Návrh pouzdra pro kamerový modul 37

Obrázek 31 Montáž kamerového modulu 38

Obrázek 32 Vstupní panel 38
Obrázek 33 Dotykový vstupní panel 38
Obrázek 34 Zařízení pro sběr teplot.....39

12 Seznam Příloh

Příloha 1 : kód kamerového modulu

Příloha 2 : kód vstupního panelu s membránovou klávesnicí

Příloha 3 : kód vstupního panelu s dotykovým displejem

Příloha 4 : kód zařízení pro sběr teplot

Příloha 5 : Webserver

Příloha 6 : Python skript pro komunikaci s MQTT

Příloha 7 : Python skript pro komunikaci s GSM modemem

Tato diplomová práce vznikla za podpory projektu Výzkum a vývoj pokročilých metod v oblasti řízení strojů a procesů - SP2020/57 financovaného Ministerstvem školství, mládeže a tělovýchovy České republiky.